Engineering Applications of Artificial Intelligence 48 (2016) 95–105

Contents lists available at ScienceDirect

Engineering Applications of Artificial Intelligence

journal homepage: www.elsevier.com/locate/engappai

Leveraging a predictive model of the workload for intelligent slot allocation schemes in energy-efficient HPC clusters

Alberto Cocaña-Fernández *, Luciano Sánchez, José Ranilla

Departamento de Informática, Universidad de Oviedo, Gijón, Spain

ARTICLE INFO

Article history:
Received 13 January 2015
Received in revised form
30 August 2015
Accepted 12 October 2015
Available online 19 November 2015

Keywords:
Energy-efficient cluster computing
Multi-criteria decision making
Evolutionary algorithms
Distal learning

ABSTRACT

A proactive mechanism to learn an efficient strategy for adaptive resource clusters is proposed. In contrast to reactive techniques, that rescale the cluster to fit the past load, a predictive strategy is adopted. The cluster incoming workload is forecasted and an optimization problem is defined whose solution is the optimal action according to a utility function. Genetic-based machine learning techniques are used, including multi-objective evolutionary algorithms under the distal supervised learning setup. Experimental evaluations show that the proactive system presented in this work improves either the energetic efficiency or the number of reconfigurations of previous approaches without a loss in the quality of service. Depending on the predictability of the workload, in real world cluster scenarios additional energy savings of up to approximately 40% were measured over the best previous approach, with a 2× factor increment in the number of reconfigurations.

1. Introduction

According to Delforge and Whitney (2014), U.S. data centres consumed 91 billion kilowatt-hours of electricity in 2013. This consumption is projected to increase to roughly 140 billion kilowatt-hours by 2020, costing $13 billion annually. The carbon footprint is also very significant and accounts for nearly 100 million metric tons of carbon pollution per year, a figure equivalent to aviation (Gartner, 2007). In particular, energy efficient operation in High Performance Computing (HPC) Clusters is a challenging issue. HPC Clusters are the main architecture for supercomputers[1] due to the high performance of commodity microprocessors and networks, to the standard tools for high performance distributed computing, and to the lower price/performance ratio (Yeo et al., 2006). Because of the ubiquity of HPC clusters, there are powerful economical and environmental incentives for developing new techniques to reduce their electrical consumption. Furthermore, improvements in consumption result in lower heat dissipation. As a side effect, the reliability of the cluster is improved and the consumption of auxiliary devices, such as power supply units, power distribution, cooling, lighting and building switchgear, is lessened as well (Emerson Network Power, 2009).

Static approaches to the problem are focused on the development of new hardware with a lower consumption, for instance low-power CPUs such as the IBM PowerPC A2 of IBM Blue Gene/Q (Haring et al., 2012; IBM Systems and Technology Group, 2011), GPUs or Intel Xeon Phi coprocessors. Dynamic approaches, on the other hand, reshape clusters to match the existing load, down-speeding or shutting down unneeded resources (Valentini et al., 2011). For example, the Dynamic Voltage and Frequency Scaling (DVFS) technique reduces CPU voltage and frequency when the CPU is idle or under-used, as shown in Hsu and Kremer (2003), Hsu and Feng (2005), Freeh et al. (2007), Lim et al. (2006), Cheng and Zeng (2011), Ge et al. (2007), Huang and Feng (2009), and Chetsa et al. (2012). In this respect, there are software frameworks that can take advantage of different cluster energy-saving features when developing energy-efficient applications, such as Alonso et al. (2012), Schubert et al. (2012), Freeh and Lowenthal (2005), Li et al. (2010), and Xian et al. (2007). Lastly, at an intermediate level between low-power hardware and energy-conscious software, there are job schedulers (Zong et al., 2007, 2010) and thermal-aware methods (Bash et al., 2007; Tang et al., 2008) that can be used in combination with software that was not designed with energy savings in mind. This intermediate-level software can scale down and up the cluster in response to changing conditions. In particular in the so-called adaptive resource clusters, the compute nodes are switched on and off on the basis of requested, idle and available resources. The ultimate purpose of this technique is to switch off all idle nodes, however the decision algorithm is not

* Corresponding author.
E-mail addresses: cocanaalberto@gmail.com (A. Cocaña-Fernández), luciano@uniovi.es (L. Sánchez), ranilla@uniovi.es (J. Ranilla).

[1] June 2014 | TOP500 Supercomputer Sites, http://www.top500.org/lists/2014/06/

http://dx.doi.org/10.1016/j.engappai.2015.10.003
0952-1976/

trivial and has competing restrictions. For instance, a high number of reconfigurations are not wanted as it might hamper the reliability of the cluster, thus a node should not be shut down if it is going to be needed shortly after.

Many different variants of the adaptive resource cluster technique exist. It was introduced in Pinheiro et al. (2001) for Load-Balancing clusters, and it was also used in Das et al. (2008), Elnozahy et al. (2002), Berral et al. (2010), Lang et al. (2010), Garcia et al. (2010), and Llamas et al. (2012) and in VMware vSphere[2] and Citrix XenServer hypervisors.[3] It has also been applied to HPC clusters in Alvarruiz et al. (2012), Dolz et al. (2011) and Xue et al. (2007). In these last works, a Knowledge based System (KBS) is used to determine the resources (e.g. the number of compute node slots) that are needed at every moment.

In all cases, the KBSs depend on certain heuristic configuration parameters such as the time of inactivity to shutdown nodes. The heuristic component of these systems is of a high importance as it governs the balance between energy savings and the number of reconfigurations. However, these parameters cannot be preset: the KBS and the parameters on which it depends must be hand tuned for the expected workload. In previous works (Cocaña Fernández et al., 2014a) we proposed to learn the heuristic parameters defined in Dolz et al. (2011) by means of a multiobjective evolutionary algorithm in a machine learning approach. The resulting system works with both OGE/SGE and PBS/TORQUE Resource Management Systems (RMS), has a good compliance with administrator preferences in all QoS, and yields reasonable energy savings and node reconfigurations.

After this, we further worked on improving the results of the KBS as the decision making mechanism of the cluster management system. In particular, we presented in Cocaña Fernández et al. (2014b) a Hybrid Genetic Fuzzy System (HGFS) that seeks the optimal rule base for the KBS by eliciting the linguistic definition of part of the aforementioned knowledge base from data, making it depend on the cluster behaviour, and having it combined with expert rules to produce a new system whose results proofed better in both linguistic interpretability and efficiency than those achieved previously in Cocaña Fernández et al. (2014a).

However, one might still wonder whether a purely reactive strategy such as the one used in both of our previous works is the best for all scenarios or, on the contrary, a significant improvement could be obtained from a predictive management approach profiting from a higher degree of adaptability to changing workload conditions. To answer this question, we present a new decision-making mechanism for the cluster management system which consists in a predictive controller based on the framework proposed in Abdelwahed et al. (2004, 2009) whose utility function is a fuzzy model learned by means of a genetic-based machine leaning (GBML) multiobjective evolutionary algorithm (MOEA) in a distal supervised learning approach (Jordan and Rumelhart, 1992). This new mechanism, along with the KBS parameter learning algorithm proposed in Cocaña Fernández et al. (2014a) and the HGFS in Cocaña Fernández et al. (2014b), is then tested in different scenarios to give a sound answer on which strategy proofs better in each case.

The remainder of the paper is as follows. Section 2 explains succinctly the architecture of the cluster management system. Section 3 summarizes the aforementioned reactive strategy. Section 4 explains the predictive controller. Section 5 shows the experimental results. Section 6 concludes the paper and discusses the future work.

[2] VMware Distributed Power Management Concepts and Use, http://www.vmware.com/files/pdf/Distributed-Power-Management-vSphere.pdf

[3] Citrix XenServer – Efficient Server Virtualization Software, http://www.citrix.com/products/xenserver/overview.html

2. Architecture

As mentioned in Cocaña Fernández et al. (2014a,b, 2015), the solution proposed consists in a service and an administration dashboard, coupled with a Database Management System, and deployed over an HPC cluster running a Resource Management System such as OGE/SGE or PBS/TORQUE.

An overview of the system is depicted in Fig. 1. The system status is monitored by the EEClusterd service, which uses a knowledge-based Decision System to perform node reconfigurations through the Power Management module. This last module switches on/off the nodes appointed by the KBS with Ethernet cards or IPMI cards (Intelligent Platform Management Interface). The EEClusterd service collects and keeps records of the RMS and of every compute node. RMS data comprises the cluster parallel environments (OGE/SGE), queues, hosts, users, and completed, queued and running jobs. Further information on the EECluster tool architecture, deployment and use can be found in Cocaña Fernández et al. (2015).

As can be seen, the decision-making mechanism is the key component of the system, since it is the one responsible for rescaling the compute nodes to match the cluster workload. Find the optimal amount of nodes that must be on at every moment given a set of preferences is no trivial problem. Because of this, both reactive and predictive strategies have been experimented with in order to achieve the best results in each scenario.

The reactive strategy consists in the reconfiguration of nodes to match the cluster status whenever the decision-making mechanism is triggered, having this status measured in terms of queued jobs, average waiting times and node idle times. Following this strategy are the KBS and HGFS explained in Section 3.

The predictive strategy consists in forecasting the cluster incoming workload and then solve an optimization problem to choose the optimal action among the permissible ones given the cluster current status and according to a machine-learned utility function. The mechanism implementing this strategy is explained in Section 4.

Observe that all of the decision-making algorithms referenced up to this point control how many slots are powered or switched off, but none of them identify the precise nodes that must be reconfigured. These nodes are selected according to its past efficiency and how long has passed since its state was changed. The rationale is to keep the most efficient nodes running for an extended time. In addition to this, nodes that failed to comply with the last order are marked, and those with earlier timestamps are preferred. The system iterates over the potentially malfunctioning nodes thus the probability of finding a repaired node is increased.

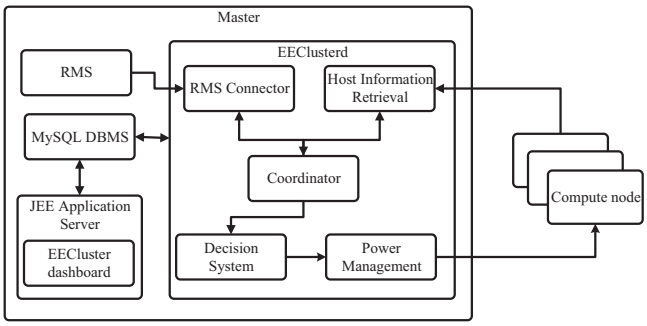

Fig. 1. System components overview.

3. Reactive decision-making through parametric KBS and hybrid genetic fuzzy systems

3.1. KBS parameter learning

The first implementation of the reactive strategy is a KBS comprised exclusively of a set of expert hand-written rules that were proposed in Dolz et al. (2011), which rely on a set of configuration parameters whose values rule the KBS operation.

These rules are as follows:

- **if** $s_{running}+s_{starting}<s_{min}$ **then** power on $(s_{min}-(s_{running}+s_{starting}))$ slots
- **if** $t_{avg}>t_{max}$ or $n_{queued}>n_{max}$ **then** power on 1 slot
- **If** $t_{avg}<t_{min}$ or $n_{queued}<n_{min}$ **then** power off 1 slot
- **for each** h in *hosts* **do**
  if $\text{idle}_h>\text{idle}_{max}$ **then** power off host h

The number of slots running and starting at a given time is respectively named $s_{running}$ and $s_{starting}$. The minimum number of slots required to run each of the queued jobs is s_{min}. The number of total slots (running and powered off) in the cluster is s_{total}. The average waiting time for the queued jobs is t_{avg}. The maximum and minimum average waiting time for the queued jobs are t_{max} and t_{min} respectively. The number of queued jobs is n_{queued}. The maximum and minimum number of queued jobs before an action is performed are n_{max} and n_{min} respectively. Finally, the time that the host h has been at idle state is called idle_h and the maximum time a host can be at idle state is idle_{max}.

If the linguistic structure of the Knowledge-based System mentioned before is not altered, each decision system can be described by the following five parameters:

$$(t_{min}, t_{max}, n_{min}, n_{max}, \text{idle}_{max}) \tag{1}$$

This Knowledge-based System can potentially adapt to any desired working mode for the cluster. However, this ability to adapt comes with the problem of actually finding the right set of values to match the desired working mode. Because of this, in Cocaña Fernández et al. (2014a), multiobjective evolutionary algorithms (MOEAs) were used to find the parameters defining the KBS, by optimizing a fitness function consisting in three conflicting criteria: the quality of service, the energy saved and the number of node reconfigurations.

3.2. Hybrid GFS

To further improve the results achieved using the previous KBS, in Cocaña Fernández et al. (2014b), was introduced a Hybrid Genetic Fuzzy System (HGFS) that is learned from data and replaces the expert-defined knowledge base with a hybrid knowledge base combining some of the expert non-fuzzy rules taken from Dolz et al. (2011) and the fuzzy rules in the form of a zero-order Takagi–Sugeno–Kang (TSK) fuzzy model (Ishibuchi et al., 2004; Takagi and Sugeno, 1985) that were learnt.

The structure of this hybrid system and can be expressed as follows:

if $s_{running}+s_{starting}<s_{min}$ **then** power on $(s_{min}-(s_{running}+s_{starting}))$ slots
if $t_{avg}>t_{max}$ or $n_{queued}>n_{max}$ **then** power on 1 slot
if $t_{avg}<t_{min}$ or $n_{queued}<n_{min}$ **then** power off 1 slot

for each h in *hosts* **do**
if idle_h is $\tilde{T}_1$ **then** off $=w_1$
if idle_h is $\tilde{T}_2$ **then** off $=w_2$
if ⋯ **then** ⋯
if idle_h is $\tilde{T}_N$ **then** off $=w_N$

All the variables defined in Section 3.1 are needed plus a few additional ones specific to the definition of the fuzzy linguistic terms. These are:

- N triangular fuzzy subsets $\tilde{T}_1,\ldots,\tilde{T}_N$ of the domain of the variables idle_h. Each triangular fuzzy number $\tilde{T}_r$ for $r=1\ldots,N$ depends on three parameters

  $(\text{left}_r, \text{center}_r, \text{right}_r)$.

  In this particular case the fuzzy sets are arranged to form a fuzzy partition (Ishibuchi et al., 2004), thus each fuzzy membership shares two parameters with the preceding element: $\text{left}_r=\text{center}_{r-1}$ and $\text{center}_r=\text{right}_{r-1}$. Therefore, the whole fuzzy partition depends on $N+2$ parameters:

  $(\text{left}_1, \text{center}_1, \text{right}_1, \text{right}_2, \ldots, \text{right}_N)$.

  Recall that idle_h is the time that the h-th host has been at idle state.
- The terms w_r, with $w_r\in[0,1]$, for $r=1\ldots,N$ that can be thought of as the degrees of truth of the asserts "if the idle time of the h-th node is $\tilde{T}_r$ then the node must be switched off".

The total number of nodes that are powered off is given by the following expression:

$$\text{Powered off nodes}=\left\lfloor\sum_{h=1}^{c}\text{TSKoutput}(\text{idle}_h)\right\rfloor \tag{2}$$

where the intermediate function $\text{TSKoutput}(t)$ is the output of the zero-order TSK fuzzy model formed by the N fuzzy rules included in the KBS, and is defined as follows:

$$\text{TSKoutput}(t)=\frac{\sum_{r=1}^{N}\tilde{T}_r(t)\cdot w_r}{\sum_{r=1}^{N}\tilde{T}_r(t)}. \tag{3}$$

Observe that a particular instance of the hybrid GFS can, therefore, be expressed as a tuple

$$(t_{min}, t_{max}, n_{min}, n_{max}, \text{left}_1, \text{center}_1, \text{right}_1, \text{right}_2, \ldots, \text{right}_N, w_1, \ldots, w_N).$$

Apart from the extra $2N+2$ parameters in the definition of an individual, the same evolutionary algorithm used in Cocaña Fernández et al. (2014a) is valid for solving the hybrid problem.

4. Proactive model

The Knowledge-based Systems introduced before both share the inner limitation of the reactive strategy: decisions of node reconfigurations are just based upon a series of values such as the number of queued jobs or the average waiting time. These values only reflect a static snapshot of the cluster status, thus disallowing these mechanisms from assessing the future consequences of each decision that make, what limits their capabilities of adaptation to volatile, although potentially stationary, scenarios with notably changing patterns of cluster activity, something rather expectable in many HPC clusters. In other words, these KBS have their operation based entirely on a set of parameters whose values were learnt upon a given cluster workload, and are not based on the actual situation of the cluster and the incoming workloads whenever the decision is made.

To improve the system results in volatile cluster scenarios, a proactive model based on the application of predictive techniques for computing systems introduced in Abdelwahed et al. (2004, 2009) is used as an alternative decision-making mechanism. It is

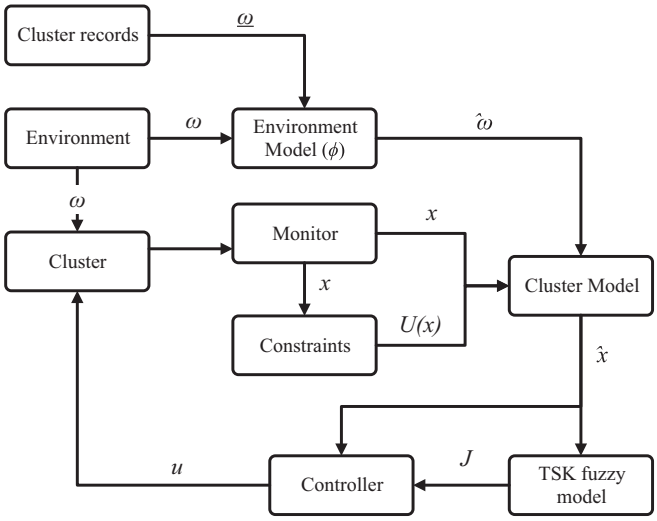

Fig. 2. Predictive controller components overview.

noteworthy that, although examples of the application of these predictive techniques can be found (see for instance Roy et al., 2011 and Bhat et al., 2006), these works have not yet been applied to the High Performance Computing (HPC) clusters addressed in this paper, which are fundamentally different from Load Balancing clusters or Grid environments in their architectures, number of concurrent requests, nature and number of resources requested and request running times.

This model transforms the cluster reconfiguration into an optimization problem of the cluster behaviour over a future temporal horizon, having this forecast by a model of both the cluster and the environment. To do so, time is split at regular intervals of $t_{interval}$ units of time, and at the beginning of each interval the control algorithm is executed resolving the optimization problem.

Fig. 2 represents the components of the controller, and the predictive control algorithm is showed in Algorithm 1.

Algorithm 1. Predictive control for an interval beginning at time k.

input: $x(k), \underline{\omega}(k-1,r)$
1: $\hat{\omega}(k) \leftarrow \phi_{k+1}(\underline{\omega}(k-1),r)$
2: **for each** $u \in U(x)$ **do**
3: $\hat{x}(k+1) \leftarrow f(x(k),u,\hat{\omega}(k))$
4: Compute utility of u based on $J(\hat{x}(k+1),u)$
5: **end for each**
output: $\underset{u}{\operatorname{argmax}}\{J(\hat{x}(k+1),u)\}$

Let k be the beginning of the current control interval, let $k+1$ be the end of the current interval and the beginning of the next one, let $x(k)$ be the cluster state and time k, let $u \in U(x)$ be a control action within the permissible actions at state x, let $w(k)$ be the actual environment and $\hat{\omega}(k)$ the forecast environment at time k, let ϕ_k be the environment forecasting model built at time k, let J be the utility function. Lastly, let f be the cluster model that runs a simulation to measure the expected outcome of taking an action u in a state $x(k)$ and with an incoming workload $\hat{\omega}(k)$ over the temporal horizon. Then, the chosen control action u at time k is the one of the allowed ones $U(x)$ for the current state $x(k)$ that gets the highest value in the utility function $J(\hat{x}(k+1),u)$, having the expected outcome $\hat{x}(k+1)$ of control action u computed by means of a simulation in the cluster model f (line 3 in Algorithm 1).

A given control action u represents the number of compute node slots that will be running after the cluster reconfiguration is done at time k. Due to obvious operating constraints, a control action cannot power on more nodes than the ones physically existing and available in the cluster. Also, since a running job cannot be halted, nodes that are currently executing jobs must never be shutdown. Because of these constraints, the set of control actions for a given state x, denoted by $U(x)$, represents the numbers of nodes that can be powered on as a result of a control action.

The rest of the section describes the environment forecasting, the utility function and the controller learning algorithm.

4.1. Environment forecasting

The cluster environment represents every external input to the system from the users in the form of jobs, and that cannot be controlled. This environment is estimated by generating synthetic workloads through the Monte Carlo simulation method using a forecasting model composed of a set of adjusted probability distributions.

Let $\hat{\omega}(k)$ be the estimated incoming workload during the control interval beginning at time k, let $\underline{\omega}(k-1,r)$ be the recorded workload of the r previous control intervals $\{\omega(k-1),\ldots,\omega(k-r-1)\}$, and let ϕ_k be the forecasting model adjusted at time k. Then, the forecast incoming workload at time k is

$$\hat{\omega}(k) = \phi_k(\underline{\omega}(k-1,r)) \tag{4}$$

If the actual environment $\omega(k)$ of the time interval that begins at time k represents the workload submitted by the cluster users in the form of n jobs, where the j-th job ($j=1\ldots n$) arrives $tarr_j$ seconds after the previous job, requests s_j slots and has a runtime of r_j seconds, then, the estimated environment $\hat{\omega}(k)$ represents $\hat{n}$ jobs where the j-th job ($j=1\ldots\hat{n}$) arrives $\hat{tarr}_j$ seconds after the previous job, requests $\hat{s}_j$ slots and has a runtime of $\hat{r}_j$ seconds. The forecasting model ϕ for this environment is formed by three models ϕ^{tarr}, ϕ^{s} and ϕ^{r}, which generate the estimated values for the arrival times, requested slots and run times of the $\hat{n}$ jobs, having this number of jobs depending on the size of the temporal horizon and the arrival time of the last job.

The values of each of these forecasting models are generated following an adjusted probability distribution. For example, if the run times of the jobs are supposed to be exponentially distributed with a rate parameter λ, then the model ϕ^r adjusted at time k generates the values:

$$\phi^r(k) = \left\{ -\frac{1}{\lambda_k}\log(U_1), -\frac{1}{\lambda_k}\log(U_2)\cdots -\frac{1}{\lambda_k}\log(U_{\hat{n}}) \right\} \tag{5}$$

where $U_1\ldots U_{\hat{n}}$ are nonzero uniform deviates.

4.2. Fuzzy utility function

As mentioned before, the control action chosen every time that the optimization problem is solved is ultimately determined by a utility function that establishes how good or bad each control action is. In other words, the utility function returns a real value which measures the expected degree of "utility" that a given control action u has for the overall cluster system. The higher the value, the more useful the action u is.

Before describing the way the utility function works, a few metrics must be defined. Since the input to this function is the expected future $x(k+1)$ resulting in executing a given action u at time k and with an expected workload $\hat{\omega}(k)$, which is done by running a simulation with the cluster model, the way the expected future state is assessed numerically so it can be used as an input to

the utility function must be established first. To do so, three values are used: the number of nodes powered on at time $k+1$ as a result of the control action, the average relation between the waiting time of the jobs and their running time u_{wdr}, and the number of reconfigured nodes u_{rn}. These last two values are computed as follows.

The average waiting time/running time is used as a quality service metric, computed as

$$u_{wdr} = \ln\left(1 + \sum_{i=1}^{\hat{n}} \frac{\text{ton}_j - \text{tsch}_j}{\text{toff}_j - \text{ton}_j}\right) \tag{6}$$

where the $\hat{n}$ jobs are the forecast workload, with the j-th job ($j = 1 \ldots \hat{n}$) arriving at time tsch_j, but starting its execution at time ton_j and stopping at time toff_j. Note that the logarithm is used as "squashing" function. As for the reconfigured nodes u_{rn}, it measures the degradation caused by the control action due to node thrashing by adding the number of nodes that are powered on and the number of nodes that are powered off:

$$u_{rn} = |\hat{x}(k+1)_{nodes} - x(k)_{nodes}| \tag{7}$$

The utility function as such is implemented using a zero-order Takagi–Sugeno–Kang (TSK) fuzzy model (Ishibuchi et al., 2004; Takagi and Sugeno, 1985), which uses u_{wdr}, $\hat{x}_{nodes}$ and u_{rn} as input values, is composed of Q rules, and whose structure can be expressed as follows:

if u_{wdr} is $\tilde{W}_1$ and $\hat{x}_{nodes}$ is $\tilde{N}_1$ and u_{rn} is $\tilde{R}_1$ **then** value $= w_1$
if u_{wdr} is $\tilde{W}_1$ and $\hat{x}_{nodes}$ is $\tilde{N}_1$ and u_{rn} is $\tilde{R}_2$ **then** value $= w_2$
if $\cdots$ **then** $\cdots$
if u_{wdr} is $\tilde{W}_1$ and $\hat{x}_{nodes}$ is $\tilde{N}_1$ and u_{rn} is $\tilde{R}_{N_3}$ **then** value $= w_{N_3}$
if u_{wdr} is $\tilde{W}_1$ and $\hat{x}_{nodes}$ is $\tilde{N}_2$ and u_{rn} is $\tilde{R}_1$ **then** value $= w_{N_3+1}$
if $\cdots$ **then** $\cdots$
If u_{wdr} is $\tilde{W}_1$ and $\hat{x}_{nodes}$ is $\tilde{N}_{N_2}$ and u_{rn} is $\tilde{R}_{N_3}$ **then** value $= w_{N_2 \times N_3}$
if u_{wdr} is $\tilde{W}_2$ and $\hat{x}_{nodes}$ is $\tilde{N}_1$ and u_{rn} is $\tilde{R}_1$ **then** value $= w_{N_2 \times N_3 + 1}$
if $\cdots$ **then** $\cdots$
If u_{wdr} is $\tilde{W}_{N_1}$ and $\hat{x}_{nodes}$ is $\tilde{N}_{N_2}$ and u_{rn} is $\tilde{R}_{N_3}$ **then** value $= w_Q$

where $\tilde{W}_1, \ldots, \tilde{W}_{N_1}$, $\tilde{N}_1, \ldots, \tilde{N}_{N_2}$ and $\tilde{R}_1, \ldots, \tilde{R}_{N_3}$ are triangular fuzzy sets forming a fuzzy partition (Ishibuchi et al., 2004) of the domain of the variables u_{wdr}, $\hat{x}_{nodes}$ and u_{rn} respectively. The partition $\tilde{W}$ has N_1 linguistic terms, $\tilde{N}$ has N_2 and $\tilde{R}$ has N_3. For instance, with $N_1 = 5$, $\tilde{T}_1$ may be "VERY LOW, "LOW", "MEDIUM", "HIGH", "VERY HIGH". The values $w_1, \ldots, w_Q$ are between 0.0 and 1.0, and represent the utility of the control action where 1.0 is the highest utility and 0.0 is the lowest. The intermediate output function is defined as follows:

$$\text{TSKOutput}(u_{wdr}, \hat{x}_{nodes}, u_{rn}) = \frac{\sum_{R_q \in S} \tilde{W}_q(u_{wdr}) \cdot \tilde{N}_q(\hat{x}_{nodes}) \cdot \tilde{R}_q(u_{rn}) \cdot w_q}{\sum_{R_q \in S} \tilde{W}_q(u_{wdr}) \cdot \tilde{N}_q(\hat{x}_{nodes}) \cdot \tilde{R}_q(u_{rn})} \tag{8}$$

Lastly, the output of the utility function can be expressed as:

$$J(\hat{x}, u) = \text{TSKoutput}(u_{wdr}, \hat{x}_{nodes}, u_{rn}) \tag{9}$$

4.3. Learning algorithm

As shown in the preceding sections, the predictive controller relies in a set of configuration parameters to determine its behaviour:

$$(t_{interval}, r, \tilde{W}_1, \ldots, \tilde{W}_{N_1}, \tilde{N}_1, \ldots, \tilde{N}_{N_2}, \tilde{R}_1, \ldots, \tilde{R}_{N_3}, w_1, \ldots, w_Q) \tag{10}$$

However, finding the right set of values to match the desired working mode for the cluster is not trivial. Leaving aside the fact that an exhaustive search is infeasible due to the large number of combinations, there is not either an optimal solution, since there are multiple conflicting objectives: the quality of service, the energy saved and the number of node reconfigurations. It is proposed that multiobjective evolutionary algorithms (MOEAs) are used to find the parameters of the predictive controller by optimizing a fitness function consisting in the three conflicting criteria. Specifically, the chosen MOEA is the Non-dominated Sorting Genetic Algorithm II (NSGA-II) (Deb et al., 2002). For a given set of n jobs, where the j-th job ($j = 1 \ldots n$) is scheduled to start at time tsch_j, but effectively starts at time ton_j and stops at time toff_j, the quality of service in a HPC cluster reflects the amount of time that each job has to wait before is assigned its requested resources. Once the job starts its execution, it will not be halted, thus we focus only on its waiting time. Because jobs do not last the same amount of time, their waiting in the queue is better expressed as a ratio considering their runtime. Finally, due to the potential existence of outlier values, the 90 percentile is used instead of average:

$$\text{QoS} = \min\left\{p : \| \left\{ j \in 1 \ldots n : \frac{\text{ton}_j - \text{tsch}_j}{\text{toff}_j - \text{ton}_j} \leq p \right\} \| > 0.9\, n \right\} \tag{11}$$

where $\|A\|$ is the cardinality of the set A.

The energy saved is measured as the sum of the amount of seconds that each node has been powered off. Let c be the number of nodes, let state(i, t) be 1 if the i-th node ($i = 1 \ldots c$) is powered at time t and 0 otherwise. Lastly, let the time scale be the lapse between $\text{tini} = \min_j\{\text{sch}_j\}$ and $\text{tend} = \max_j\{\text{toff}_j\}$. Then,

$$\text{Energy saved} = c \cdot (\text{tend} - \text{tini}) - \sum_{i=1}^{c} \int_{\text{tini}}^{\text{tend}} \text{state}(i, t)\, \text{d}t. \tag{12}$$

The node reconfigurations are the number of times that a node has been powered on or off. Let $\text{nd}(i)$ the number of discontinuities of the function state(i, t) in the time interval $t \in (\text{tini}, \text{tend})$:

$$\text{Reconfigured nodes} = \sum_{i=1}^{c} \text{nd}(i) \tag{13}$$

Although MOEAs can address the problem of finding Pareto-optimal solutions, the learning problem has an additional complication: each solution found by the NSGA-II algorithm must be tested in a cluster environment to measure the results. This is known as the distal supervised learning problem (Jordan and Rumelhart, 1992), where the learning algorithm must control indirectly the cluster outcome (distal variables) through the instances of the predictive controller (proximal variables) having the outcome fitness as a feedback to guide the learning process. This can be seen in Fig. 3, where the NSGA-II learning algorithm produces predictive controller instances, which are then tested in a cluster simulator with a given workload to compute their fitness value. It is also remarked that in the provided results the membership functions $\tilde{W}_1, \ldots, \tilde{W}_{N_1}, \tilde{N}_1, \ldots, \tilde{N}_{N_2}, \tilde{R}_1, \ldots, \tilde{R}_{N_3}$ will not be

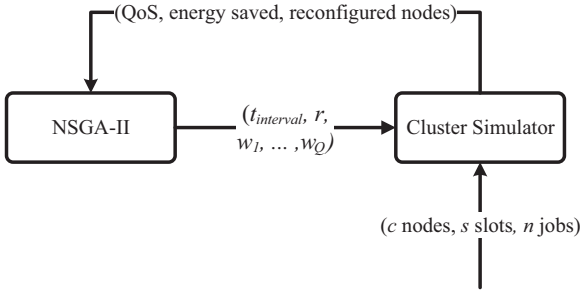

Fig. 3. Distal supervised learning of the predictive controller.

adjusted but a uniform partition is defined instead. This is not a fundamental limitation, since any change in the membership function of these sets could be compensated by the corresponding modification in the weights w_q.

5. Experimental results

Since HPC clusters may exhibit very different patterns of activity depending on their applications, a sound answer about which of the described decision-making mechanism works better involves thorough testing of different scenarios building together a significant representation of HPC clusters. To do so, four cluster scenarios have been defined in terms of synthetic cluster workloads characterized by the probability distributions of both job arrival rates and run times. The run times are distributed exponentially with rate $\lambda = 10^{-5}$ s in all scenarios, and the arrivals follow a Poisson process with the rate values shown in Table 1.

Essentially, scenario 1 depicts a cluster environment where jobs are submitted following an extremely regular arrival pattern where each hour of the year shares the same arrival rate. Scenario 2 also maintains regularity on a weekly basis, but distinguishes clearly between working hours, non-working hours and weekends, adjusting the arrival rate accordingly to each hour range. Scenario 3 adds a significant variation in arrival rates between consecutive weeks, and scenario 4 increases the degree of variation between weeks.

In addition, experiments with actual workloads from the Scientific Modelling Cluster of the University of Oviedo (CMS) spanning 22 months, with a total of 2907 jobs, were performed. This real world cluster, consisting of three independent computing clusters and five transversal queues using PBS as Resource Management System (RMS), can accurately show a very common activity pattern in most HPC clusters.

Similar to the experimentation in our previous work, a cluster simulator has been developed for both training and testing, so that every model can be evaluated in the three criteria of the fitness function.

The three decision-making mechanisms described in Section 3 and 4, along with mechanisms proposed by other authors, have been tested using this simulator in combination with the five workloads described before:

1. A basic model (labelled "Single rule") consisting in the allocation of as many compute node slots as are required to run all queued jobs, shutting down every idle node whenever the decision mechanism is triggered.
2. The rule model proposed in Dolz et al. (2011) (labelled as "EnergySaving ($t_{min}, t_{max}, n_{min}, n_{max}, \text{idle}_{max}$)") with multiple configurations and its parameters hand tuned by the administrator.
3. The rule model proposed in Alvarruiz et al. (2012), labelled as "CLUES ($\text{extra}_{slots}, \text{idle}_{max}$)", where extra_{slots} represents the number of extra slots that are powered on whenever additional slots are required to serve the current workload. Similar to the previous mechanism, multiple configurations were tested with their parameters hand tuned by the administrator.
4. The rule model proposed in Dolz et al. (2011) (labelled as "EnergySaving KBS NSGA-II") with the learning mechanism proposed in Cocaña Fernández et al. (2014a)
5. The hybrid GFS proposed in Cocaña Fernández et al., 2014b (labelled as "Hybrid GFS NSGA-II")
6. The predictive controller proposed in this paper, labelled as "Predictive controller (N_1, N_2, N_3)", with a different number of linguistic terms in each partition.

The holdout method was used for validation, with a 50–25–25% split in training, validation and test.

The administrator preferences for the experiment are based upon a lexicographic ordering of the three criteria: the administrator always seeks the best QoS and the amount of energy saved is used only to break ties in QoS. In turn, the number of reconfigurations also serves to break ties in QoS and energy saving.

The experiment results are shown in the following tables in terms of QoS, energy saved and node reconfigurations, and in the following charts in terms of the cluster simulation traces, which reveal the evolution over time of the aggregated requested slots by the jobs and the slots powered on by each mechanism. In particular, results for the scenario 1 are displayed in Table 2 and in Fig. 4, scenario 2 in Table 3 and in Fig. 5, scenario 3 in Table 4 and in Fig. 6, and scenario 4 in Table 5 and in Fig. 7. Lastly, results obtained for the CMS cluster recorded workloads are displayed in Table 6 and in Fig. 8.

It is remarked that the "QoS" column displays the 90 percentile (see Eq. (11)) thus a value of zero means that more than 90% of tasks were not delayed.

These results show that excessively simple mechanisms, such as the one labelled as "Single rule" do not perform well, negatively impacting cluster service quality, and neither can be tuned to suit administrator preferences due to the lack of configuration parameters. Similarly, mechanisms that despite relying on parameters to rule their function require these to be configured manually often perform poorly, impacting QoS and causing node thrashing. This is ultimately due to the complex task of finding the right set of values to comply with administrator preferences because of the large number of combinations.

Results also show that when the cluster workload exhibits a pattern of high arrival regularity, such as in scenario 1 and relatively in scenario 2, the decision-making mechanisms following a reactive strategy tend to achieve significantly better results in energy saving than the one following a predictive strategy. In this type of scenarios the distance between local workload peaks is very short, and so is the distance between peaks and valleys. The best approach is possibly to do minor cluster adjustments over short periods of time to save energy during these short-duration valleys. The reactive strategy is well suited to this situation. This also explains why the predictive controller barely reconfigures the

Table 1
Poisson process of job arrivals in each scenario.

Scenario	Day of week	Hour range	Week of year	λ value (s)
1	All	All	All	2×10^{-4}
	Mon–Fri	8:00–20:00	All	2×10^{-4}
2	Sat–Sun	8:00–20:00	All	2×10^{-5}
	Mon–Sun	20:00–8:00	All	10^{-5}
3	Mon– Fri	8:00–20:00	$w\ \%5 = 0$	10^{-4}
			$w\ \%\ 5 = 1$	2×10^{-4}
			$w\ \%\ 5 = 2$	5×10^{-4}
			$w\ \%\ 5 = 3$	5×10^{-4}
			$w\ \%\ 5 = 4$	2×10^{-4}
	Mon–Sun	20:00–8:00	All	2×10^{-5}
	Mon–Fri	8:00–20:00	All	10^{-5}
4	Mon– Fri	8:00–20:00	$w\ \%\ 5 = 0$	10^{-4}
			$w\ \%\ 5 = 1$	10^{-4}
			$w\ \%\ 5 = 2$	5×10^{-4}
			$w\ \%\ 5 = 3$	5×10^{-4}
			$w\ \%\ 5 = 4$	10^{-4}
	Mon–Sun	20:00–8:00	All	2×10^{-5}
	Mon–Fri	8:00–20:00	All	10^{-5}

Table 2
Experiment results for the test set of the scenario 1.

Decision-making mechanism	Scenario 1 test set		
	QoS	Energy saved(s)	Reconfigurations
Single rule	4.49E−02	2.09E+08	5931
EnergySaving (0, 60, 0, 5, 3600)	2.74E−02	2.02E+08	3639
EnergySaving (0, 300, 0, 10, 3600)	4.24E−02	2.02E+08	3523
EnergySaving (0, 60, 0, 5, 7200)	2.14E−02	1.96E+08	2757
EnergySaving (0, 60, 0, 0, 14 400)	1.31E−02	1.88E+08	1968
CLUES (0, 3600)	2.72E−02	2.02E+08	3721
CLUES (0, 7200)	1.99E−02	1.96E+08	2801
CLUES (0, 14 400)	1.31E−02	1.88E+08	1962
CLUES (2, 3600)	1.77E−02	1.93E+08	7157
CLUES (2, 7200)	1.05E−02	1.87E+08	4609
CLUES (2, 14 400)	5.81E−03	1.79E+08	2827
EnergySaving KBS NSGA-II	0.00E+00	1.34E+08	473
Hybrid GFS NSGA-II	0.00E+00	1.49E+08	487
Predictive controller (3, 2, 2)	0.00E+00	9.32E+07	6
Predictive controller (3, 3, 3)	0.00E+00	9.31E+07	6
Predictive controller (4, 3, 3)	0.00E+00	9.32E+07	6
Predictive controller (5, 3, 3)	0.00E+00	9.24E+07	6
Predictive controller (5, 4, 4)	0.00E+00	9.24E+07	6

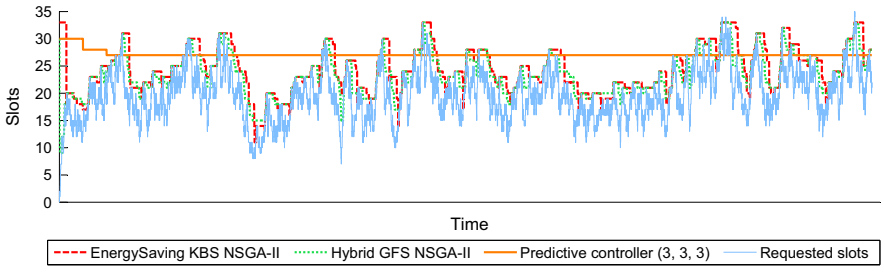

Fig. 4. Cluster simulation trace for the test set of the scenario 1.

Table 3
Experiment results for the test set of the scenario 2.

Decision-making mechanism	Scenario 2 test set		
	QoS	Energy saved(s)	Reconfigurations
Single rule	4.18E−02	4.03E+08	2390
EnergySaving (0, 60, 0, 5, 3600)	3.61E−02	3.99E+08	1962
EnergySaving (0, 300, 0, 10, 3600)	5.32E−02	3.99E+08	1944
EnergySaving (0, 60, 0, 5, 7200)	3.26E−02	3.96E+08	1754
EnergySaving (0, 60, 0, 0, 14 400)	2.48E−02	3.91E+08	1502
CLUES (0, 3600)	3.15E−02	3.99E+08	1972
CLUES (0, 7200)	2.93E−02	3.96E+08	1758
CLUES (0, 14 400)	2.51E−02	3.91E+08	1504
CLUES (2, 3600)	2.36E−02	3.95E+08	3900
CLUES (2, 7200)	1.60E−02	3.91E+08	2876
CLUES (2, 14 400)	1.20E−02	3.84E+08	2164
EnergySaving KBS NSGA-II	0.00E+00	2.92E+08	236
Hybrid GFS NSGA-II	0.00E+00	3.14E+08	218
Predictive controller (3, 2, 2)	0.00E+00	2.78E+08	18
Predictive controller (3, 3, 3)	0.00E+00	2.94E+08	20
Predictive controller (4, 3, 3)	0.00E+00	2.83E+08	18
Predictive controller (5, 3, 3)	0.00E+00	2.83E+08	18
Predictive controller (5, 4, 4)	0.00E+00	2.85E+08	27

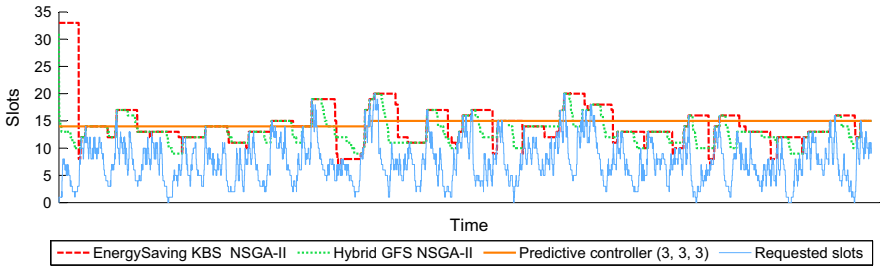

Fig. 5. Cluster simulation trace for the test set of the scenario 2.

Table 4
Experiment results for the test set of the scenario 3.

Decision-making mechanism	Scenario 3 test set		
	QoS	Energy saved(s)	Reconfigurations
Single rule	4.53E−02	3.49E+08	3394
EnergySaving (0, 60, 0, 5, 3600)	3.89E−02	3.44E+08	2586
EnergySaving (0, 300, 0, 10, 3600)	5.95E−02	3.44E+08	2552
EnergySaving (0, 60, 0, 5, 7200)	3.51E−02	3.40E+08	2325
EnergySaving (0, 60, 0, 0, 14 400)	2.88E−02	3.34E+08	2077
CLUES (0, 3600)	3.99E−02	3.44E+08	2626
CLUES (0, 7200)	3.51E−02	3.40E+08	2343
CLUES (0, 14 400)	3.16E−02	3.33E+08	2081
CLUES (2, 3600)	2.52E−02	3.39E+08	4442
CLUES (2, 7200)	1.91E−02	3.35E+08	3327
CLUES (2, 14 400)	1.47E−02	3.27E+08	2595
EnergySaving KBS NSGA-II	0.00E+00	1.19E+08	202
Hybrid GFS NSGA-II	0.00E+00	1.46E+08	194
Predictive controller (3, 2, 2)	0.00E+00	1.29E+08	175
Predictive controller (3, 3, 3)	0.00E+00	1.47E+08	337
Predictive controller (4, 3, 3)	0.00E+00	1.76E+08	338
Predictive controller (5, 3, 3)	0.00E+00	1.20E+08	479
Predictive controller (5, 4, 4)	0.00E+00	1.14E+08	155

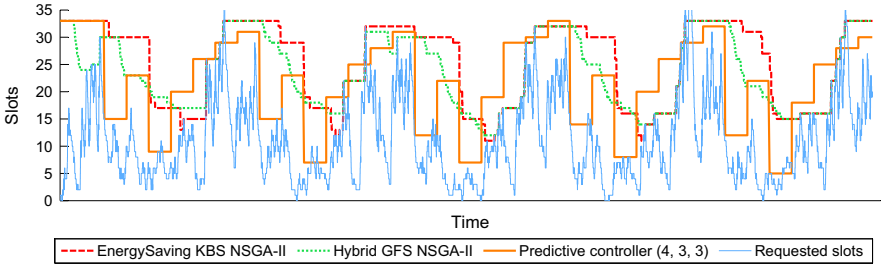

Fig. 6. Cluster simulation trace for the test set of the scenario 3.

cluster: the short duration of the valleys makes it very difficult for the controller to reconfigure the cluster over the rather long temporal horizons and save energy with no impact on QoS.

On the other hand, in cluster scenarios that exhibit a high degree of variation in the job arrival rates, such as scenarios 3 and 4, the predictive controller achieves better results that the reactive ones. Similar to the previous case, the key is both the duration of the workload valleys and the size of the local peaks, and how these values in scenarios 3 and 4 fit well the longer temporal horizons of the predictive controller. In particular, a comparison between results in scenarios 3 and 4 shows that the longer the valleys and the bigger the distance between peaks and valleys are, the better results are obtained by the predictive controller compared to the reactive ones. The reason for the bad performance of the reactive controllers in these scenarios, specially the expert-defined KBS, is that these mechanisms rely heavily on the time that the hosts have been at idle state to save energy. This is a problem here because, in order to assure good service quality whenever the load swiftly increases, the idle values are set too high to allow good energy saving or, otherwise, a very negative impact on QoS would occur as the workload grows. Regarding the results obtained in a real world cluster such as the CMS, these are very similar to the ones in scenarios 3 and 4, as could be expected due to the resemblance of the CMS activity pattern to these scenarios (see Fig. 8).

Lastly, it should also be noted that the Hybrid GFS always obtains better energy savings that the expert-defined KBS, what is achieved thanks to the higher flexibility of the fuzzy rule base regarding the host idle times.

Table 5
Experiment results for the test set of the scenario 4.

Decision-making mechanism	Scenario 4 test set		
	QoS	Energy saved(s)	Reconfigurations
Single rule	4.24E−02	3.66E+08	3031
EnergySaving (0, 60, 0, 5, 3600)	3.42E−02	3.62E+08	2275
EnergySaving (0, 300, 0, 10, 3600)	5.19E−02	3.62E+08	2255
EnergySaving (0, 60, 0, 5, 7200)	3.12E−02	3.58E+08	2022
EnergySaving (0, 60, 0, 0, 14 400)	2.61E−02	3.52E+08	1820
CLUES (0, 3600)	3.29E−02	3.61E+08	2289
CLUES (0, 7200)	2.92E−02	3.58E+08	2028
CLUES (0, 14 400)	2.81E−02	3.52E+08	1818
CLUES (2, 3600)	2.19E−02	3.57E+08	3923
CLUES (2, 7200)	1.65E−02	3.53E+08	3048
CLUES (2, 14 400)	1.24E−02	3.45E+08	2395
EnergySaving KBS NSGA-II	0.00E+00	3.61E+07	8
Hybrid GFS NSGA-II	0.00E+00	1.31E+08	173
Predictive controller (3, 2, 2)	0.00E+00	1.69E+08	414
Predictive controller (3, 3, 3)	0.00E+00	1.78E+08	393
Predictive controller (4, 3, 3)	0.00E+00	1.62E+08	252
Predictive controller (5, 3, 3)	0.00E+00	1.33E+08	273
Predictive controller (5, 4, 4)	0.00E+00	1.01E+08	52

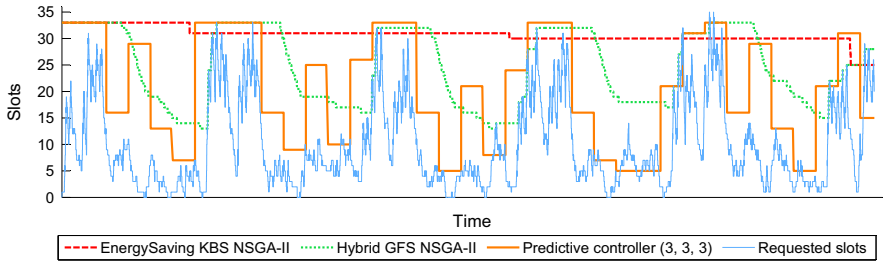

Fig. 7. Cluster simulation trace for the test set of the scenario 4.

Table 6
Experiment results for the test set of the CMS cluster workload records.

Decision-making mechanism	CMS cluster test set		
	QoS	Energy saved (s)	Reconfigurations
Single rule	8.02E+01	4.22E+08	2504
EnergySaving (0, 60, 0, 5, 3600)	4.86E+01	4.25E+08	1538
EnergySaving (0, 300, 0, 10, 3600)	7.74E+01	4.26E+08	1512
EnergySaving (0, 60, 0, 5, 7200)	2.23E+01	4.23E+08	1386
EnergySaving (0, 60, 0, 0, 14 400)	2.92E+00	4.19E+08	1216
CLUES (0, 3600)	3.43E+01	4.20E+08	2724
CLUES (0, 7200)	1.28E+01	4.16E+08	2308
CLUES (0, 14 400)	4.24E+00	4.11E+08	1828
CLUES (2, 3600)	2.60E+01	4.18E+08	3810
CLUES (2, 7200)	5.95E+00	4.14E+08	2942
CLUES (2, 14 400)	2.66E+00	4.07E+08	2366
EnergySaving KBS NSGA-II	0.00E+00	1.88E+08	47
Hybrid GFS NSGA-II	0.00E+00	2.41E+08	42
Predictive controller (3, 2, 2)	0.00E+00	2.01E+08	77
Predictive controller (3, 3, 3)	0.00E+00	3.00E+08	96
Predictive controller (4, 3, 3)	0.00E+00	2.78E+08	89
Predictive controller (5, 3, 3)	0.00E+00	2.72E+08	70
Predictive controller (5, 4, 4)	0.00E+00	2.03E+08	95

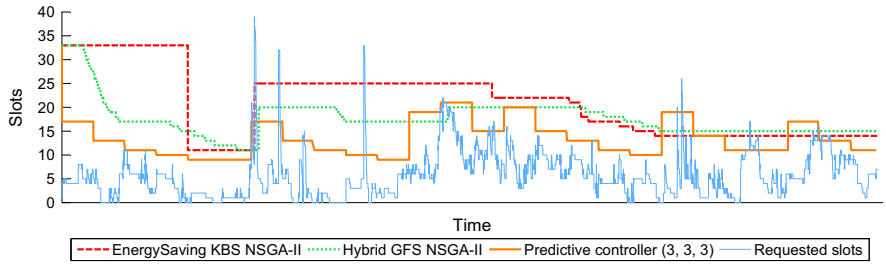

Fig. 8. Cluster simulation trace for the test set of the CMS cluster workload records.

6. Concluding remarks

Adaptive resource clusters are an efficient method for reducing electrical consumption, however this technique depends on a decision algorithm that has conflicting restrictions. Since a high number of reconfigurations are not desired, a node should not be shut down if it is going to be needed shortly after. Reactive techniques fulfill this objective by means of heuristics such as imposing delays before a node is stopped after a valley in the workload, or enforcing a minimum uptime for each functioning node. However, the best balance between consumption and reconfigurations is achieved with the proactive model described in this paper. The proposed strategy consists in forecasting the cluster incoming workload and then solving an optimization problem to choose the optimal action according to a fuzzy utility function. Specific genetic-based machine learning techniques were deployed that consist of mutiobjective evolutionary algorithms under the distal supervised learning setup. Empirical results prove that reactive systems tend to consume less energy in scenarios with a constant job arrival rate; nonetheless, the proactive system presented in this work achieves the highest energetic efficiency when the workload is not quite regular, as happens in most of practical scenarios.

Acknowledgements

This work has been partially supported by Ministerio de Economía y Competitividad from Spain/FEDER under Grants TEC2012-38142-C04-04, TEC2015-67387-C4-3-R and TIN2014-56967-R and by the Regional Ministry of the Principality of Asturias under Grant FC-15-GRUPIN14-073.

References

Abdelwahed, S., Bai, J., Su, R., Kandasamy, N., 2009. On the application of predictive control techniques for adaptive performance management of computing systems. IEEE Trans. Netw. Serv. Manag. 6 (December (4)), 212–225, URL ⟨http://ieeexplore.ieee.org/articleDetails.jsp?arnumber=5374030⟩.

Abdelwahed, S., Kandasamy, N., Neema, S., 2004. A control-based framework for self-managing distributed computing systems. In: Proceedings of the First ACM SIGSOFT Workshop on Self-Managed Systems—WOSS '04. ACM Press, New York, NY, USA, pp. 3–7, October. URL ⟨http://dl.acm.org/citation.cfm?id=1075405.1075406⟩.

Alonso, P., Badia, R.M., Labarta, J., Barreda, M., Dolz, M.F., Mayo, R., Quintana-Orti, E. S., Reyes, R., 2012. Tools for power-energy modelling and analysis of parallel scientific applications. In: 2012 41st International Conference on Parallel Processing. IEEE, Pittsburgh, PA, pp. 420–429, September. URL⟨http://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=6337603⟩.

Alvarruiz, F., de Alfonso, C., Caballer, M., Hernández, V., 2012. An energy manager for high performance computer clusters. In: 2012 IEEE 10th International Symposium on Parallel and Distributed Processing with Applications. IEEE, Leganés, Madrid, Spain, pp. 231–238, July. URL ⟨http://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=6280297⟩.

Bash, C., Forman, G., 2007. Cool Job Allocation: Measuring the Power Savings of Placing Jobs at Cooling-Efficient Locations in the Data Center. USENIX Association, pp. 29:1–29:6 June. URL ⟨http://dl.acm.org/citation.cfm?id=1364385.1364414⟩.

Berral, J.L., Goiri, I.N., Nou, R., Julià, F., Guitart, J., Gavaldà, R., Torres, J., 2010. Towards energy-aware scheduling in data centers using machine learning. In: Proceedings of the First International Conference on Energy-Efficient Computing and Networking—e-Energy '10. ACM Press, New York, NY, USA, p. 215, April. URL ⟨http://dl.acm.org/citation.cfm?id=1791314.1791349⟩.

Bhat, V., Parashar, M., Khandekar, M., Kandasamy, N., Klasky, S., 2006. A self-managing wide-area data streaming service using model-based online control. In: Proceedings of the Seventh IEEE/ACM International Conference on Grid Computing. GRID '06. IEEE Computer Society, Washington, DC, USA, pp. 176–183. http://dx.doi.org/10.1109/ICGRID.2006.311013.

Cheng, Y., Zeng, Y., 2011. Automatic energy status controlling with dynamic voltage scaling in power-aware high performance computing cluster. In: 2011 12th International Conference on Parallel and Distributed Computing, Applications and Technologies. IEEE, Gwangju, Korea, pp. 412–416, October. URL ⟨http://ieeexplore.ieee.org/articleDetails.jsp?arnumber=6118547⟩.

Chetsa, G.L.T., Lefrvre, L., Pierson, J.-M., Stolf, P., Da Costa, G., 2012. A runtime framework for energy efficient HPC systems without a priori knowledge of applications. In: 2012 IEEE 18th International Conference on Parallel and Distributed Systems. IEEE, Singapore, pp. 660–667, December. URL ⟨http://ieeexplore.ieee.org/articleDetails.jsp?arnumber=6413638⟩.

Cocaña Fernández, A., Ranilla, J., Sánchez, L., 2014a. Energy-efficient allocation of computing node slots in HPC clusters through evolutionary multi-criteria decision making. In: Proceedings of the 14th International Conference on Computational and Mathematical Methods in Science and Engineering, CMMSE 2014, pp. 318–330.

Cocaña Fernández, A., Ranilla, J., Sánchez, L., 2014b. Energy-efficient allocation of computing node slots in HPC clusters through parameter learning and hybrid genetic fuzzy system modeling. J. Supercomput. (October). URL ⟨http://link.springer.com/10.1007/s11227-014-1320-9⟩.

Cocaña Fernández, A., Sánchez, L., Ranilla, J., 2015. A software tool to efficiently manage the energy consumption of HPC clusters. In: 2015 IEEE International Conference on Fuzzy Systems (FUZZ-IEEE 2015).

Das, R., Kephart, J.O., Lefurgy, C., Tesauro, G., Levine, D.W., Chan, H., 2008. Autonomic Multi-Agent Management of Power and Performance in Data Centers, pp. 107–114, May. URL ⟨http://dl.acm.org/citation.cfm?id=1402795.1402816⟩.

Deb, K., Pratap, A., Agarwal, S., Meyarivan, T., 2002. A fast and elitist multiobjective genetic algorithm: NSGA-II. IEEE Trans. Evol. Comput. 6 (April(2)), 182–197. URL ⟨http://ieeexplore.ieee.org/articleDetails.jsp?arnumber=996017⟩.

Delforge, P., Whitney, J., 2014. Issue Paper: Data Center Efficiency Assessment Scaling up Energy Efficiency Across the Data Center Industry: Evaluating Key Drivers and Barriers. Technical Report, Natural Resources Defense Council (NRDC). URL ⟨http://www.nrdc.org/energy/files/data-center-efficiency-assessment-IP.pdf⟩.

Dolz, M.F., Fernández, J.C., Iserte, S., Mayo, R., Quintana-Ortí, E.S., Cotallo, M. E., Díaz, G., 2011. EnergySaving cluster experience in CETA-CIEMAT. In: The Fifth Iberian GRID Infrastructure Conference, Santander.

Elnozahy, E.N., Kistler, M., Rajamony, R., 2002. Energy-Efficient Server Clusters, pp. 179–197, February. URL ⟨http://dl.acm.org/citation.cfm?id=1766991.1767007⟩.

Emerson Network Power, 2009. Energy Logic: Reducing Data Center Energy Consumption by Creating Savings that Cascade Across Systems. Technical Report. URL ⟨https://www.cisco.com/web/partners/downloads/765/other/Energy_Logic_Reducing_Data_Center_Energy_Consumption.pdf⟩.

Freeh, V.W., Lowenthal, D.K., 2005. Using multiple energy gears in MPI programs on a power-scalable cluster. In: Proceedings of the Tenth ACM SIGPLAN Symposium on Principles and Practice of Parallel Programming—PPoPP '05. ACM Press, New York, NY, USA, pp. 164–173, June. URL ⟨http://dl.acm.org/citation.cfm?id=1065944.1065967⟩.

Freeh, V.W., Lowenthal, D.K., Pan, F., Kappiah, N., Springer, R., Rountree, B.L., Femal, M.E., 2007. Analyzing the energy-time trade-off in high-performance computing applications. IEEE Trans. Parallel Distrib. Syst. 18 (June (6)), 835–848, URL ⟨http://dl.acm.org/citation.cfm?id=1263127.1263246⟩.

Garcia, D.F., Entrialgo, J., Garcia, J., Garcia, M., 2010. A self-managing strategy for balancing response time and power consumption in heterogeneous server clusters. In: 2010 International Conference on Electronics and Information Engineering, vol. 1. IEEE, pp. V1-537–V1-541, August. URL ⟨http://ieeexplore.ieee.org/articleDetails.jsp?arnumber=5559691⟩.

Gartner, 2007. Gartner Estimates ICT Industry Accounts for 2 Percent of Global CO2 Emissions. URL ⟨http://www.gartner.com/newsroom/id/503867⟩.

Ge, R., Feng, X., Feng, W.-c., Cameron, K.W., 2007. CPU MISER: a performance-directed, run-time system for power-aware clusters. In: 2007 International Conference on Parallel Processing (ICPP 2007). IEEE, p. 18, September. URL ⟨http://ieeexplore.ieee.org/articleDetails.jsp?arnumber=4343825⟩.

Haring, R., Ohmacht, M., Fox, T., Gschwind, M., Satterfield, D., Sugavanam, K., Coteus, P., Heidelberger, P., Blumrich, M., Wisniewski, R., alan Gara, Chiu, G., Boyle, P., Chist, N., Kim, C., Mar. 2012. The IBM blue gene/Q compute chip. IEEE Micro 32 (2), 48–60. URL ⟨http://ieeexplore.ieee.org/articleDetails.jsp?arnumber=6109225⟩.

Hsu, C.-H., Feng, W.-c., 2005. A power-aware run-time system for high-performance computing. In: ACM/IEEE SC 2005 Conference (SC'05). IEEE, Washington, DC, USA, p. 1. URL ⟨http://ieeexplore.ieee.org/articleDetails.jsp?arnumber=1559953⟩.

Hsu, C.-H., Kremer, U., 2003. The design, implementation, and evaluation of a compiler algorithm for CPU energy reduction. ACM SIGPLAN Not. 38 (May (5)), 38, URL ⟨http://dl.acm.org/citation.cfm?id=780822.781137⟩.

Huang, S., Feng, W., 2009. Energy-efficient cluster computing via accurate workload characterization. In: 2009 Ninth IEEE/ACM International Symposium on Cluster Computing and the Grid. IEEE, Shanghai, China, pp. 68–75. URL ⟨http://ieeexplore.ieee.org/articleDetails.jsp?arnumber=5071856⟩.

IBM Systems and Technology Group, 2011. IBM System Blue Gene/Q. Technical Report, IBM, Somers, NY. URL ⟨http://www.fz-juelich.de/SharedDocs/Downloads/IAS/JSC/EN/JUQUEEN/BGQIBMDataSheet.pdf?__blob=publicationFile⟩.

Ishibuchi, H., Nakashima, T., Nii, M., 2004. Classification and Modeling with Linguistic Information Granules: Advanced Approaches to Linguistic Data Mining (Advanced Information Processing), November. URL ⟨http://dl.acm.org/citation.cfm?id=1044904⟩.

Jordan, M., Rumelhart, D.E., 1992. Forward models: supervised learning with a distal teacher. Cogn. Sci. 16 (September (3)), 307–354 URL ⟨http://www.sciencedirect.com/science/article/pii/036402139290036T⟩.

Lang, W., Patel, J.M., Naughton, J.F., 2010. On energy management, load balancing and replication. ACM SIGMOD Rec. 38 (June (4)), 35–42 URL ⟨http://dl.acm.org/citation.cfm?id=1815948.1815956⟩.

Li, D., Nikolopoulos, D.S., Cameron, K., de Supinski, B.R., Schulz, M., 2010. Power-aware MPI task aggregation prediction for high-end computing systems. In: 2010 IEEE International Symposium on Parallel& Distributed Processing (IPDPS). IEEE, Atlanta, GA, pp. 1–12. URL ⟨http://ieeexplore.ieee.org/articleDetails.jsp?arnumber=5470464⟩.

Lim, M., Freeh, V., Lowenthal, D., 2006. Adaptive, transparent frequency and voltage scaling of communication phases in mpi programs. In: ACM/IEEE SC 2006 Conference (SC'06). IEEE, Tampa, FL, p. 14, November. URL ⟨http://ieeexplore.ieee.org/articleDetails.jsp?arnumber=4090188⟩.

Llamas, R.M., Garcia, D.F., Entrialgo, J., 2012. A technique for self-optimizing scalable and dependable server clusters under qos constraints. In: 2012 IEEE 11th International Symposium on Network Computing and Applications. IEEE, pp. 61–66, August. URL ⟨http://ieeexplore.ieee.org/articleDetails.jsp?arnumber=6299127⟩.

Pinheiro, E., Bianchini, R., Carrera, E. V., Heath, T., 2001. Load balancing and unbalancing for power and performance in cluster-based systems. In: Workshop on Compilers and Operating Systems for Low Power, vol. 180, Barcelona, Spain, pp. 182–195.

Roy, N., Dubey, A., Gokhale, A., 2011. Efficient autoscaling in the cloud using predictive models for workload forecasting. In: 2011 IEEE International Conference on Cloud Computing (CLOUD), pp. 500–507, July.

Schubert, S., Kostic, D., Zwaenepoel, W., Shin, K.G., 2012. Profiling software for energy consumption. In: 2012 IEEE International Conference on Green Computing and Communications. IEEE, Besancon, France, pp. 515–522, November. URL ⟨http://ieeexplore.ieee.org/articleDetails.jsp?arnumber=6468359⟩.

Takagi, T., Sugeno, M., 1985. Fuzzy identification of systems and its applications to modeling and control. IEEE Trans. Syst. Man Cybern. 15 (January (1)), 116–132, URL ⟨http://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=6313399⟩.

Tang, Q., Gupta, S.K.S, Varsamopoulos, G., 2008. Energy-efficient thermal-aware task scheduling for homogeneous high-performance computing data centers: a cyber-physical approach. IEEE Trans. Parallel Distrib. Syst. 19 (11), 1458–1472, November. URL ⟨http://ieeexplore.ieee.org/articleDetails.jsp?arnumber=4553706⟩.

Valentini, G.L., Lassonde, W., Khan, S.U., Min-Allah, N., Madani, S.A., Li, J., Zhang, L., Wang, L., Ghani, N., Kolodziej, J., Li, H., Zomaya, A.Y., Xu, C.-Z., Balaji, P., Vishnu, A., Pinel, F., Pecero, J.E., Kliazovich, D., Bouvry, P., 2011. An overview of energy efficiency techniques in cluster computing systems. Clust. Comput. 16 (September (1)), 3–15 URL ⟨http://link.springer.com/10.1007/s10586-011-0171-x⟩.

Xian, C., Lu, Y.-H., Li, Z., 2007. A programming environment with runtime energy characterization for energy-aware applications. In: Proceedings of the 2007 international symposium on Low power electronics and design—ISLPED '07. ACM Press, New York, New York, USA, pp. 141–146, August. URL ⟨http://dl.acm.org/citation.cfm?id=1283780.1283811⟩.

Xue, Z., Dong, X., Ma, S., Fan, S., Mei, Y., 2007. An Energy-efficient management mechanism for large-scale server clusters. In: The Second IEEE Asia-Pacific Service Computing Conference (APSCC 2007). IEEE, Tsukuba Science City, Japan, pp. 509–516, December. URL ⟨http://ieeexplore.ieee.org/articleDetails.jsp?arnumber=4414502⟩.

Yeo, CheeShin, Buyya, Rajkumar, Pourreza, Hossein, Eskicioglu, Rasit, Graham, Peter, Sommers, F., 2006. Cluster computing: high-performance, high-availability, and high-throughput processing on a network of computers. In: Zomaya, A. (Ed.), Handbook of Nature-Inspired and Innovative Computing. Springer, USA, pp. 521–551. http://dx.doi.org/10.1007/0-387-27705-6_16.

Zong, Z., Nijim, M., Manzanares, A., Qin, X., 2007. Energy efficient scheduling for parallel applications on mobile clusters. Clust. Comput. 11 (November (1)), 91–113, URL ⟨http://dl.acm.org/citation.cfm?id=1349620.1349631⟩.

Zong, Z., Ruan, X., Manzanares, A., Bellam, K., Qin, X., 2010. Improving energy-efficiency of computational grids via scheduling. In: Antonopoulos, N., Exarchakos, G., Li, M., Liotta, A. (Eds.), Handbook of Research on P2P and Grid Systems for Service-Oriented Computing. IGI Global (Chapter 22), January. URL ⟨http://www.igi-global.com/chapter/improving-energy-efficiency-computational-grids/40816/⟩.